



School Of Engineering

Electronics and Communication Engineering

TITLE OF THE PROJECT :

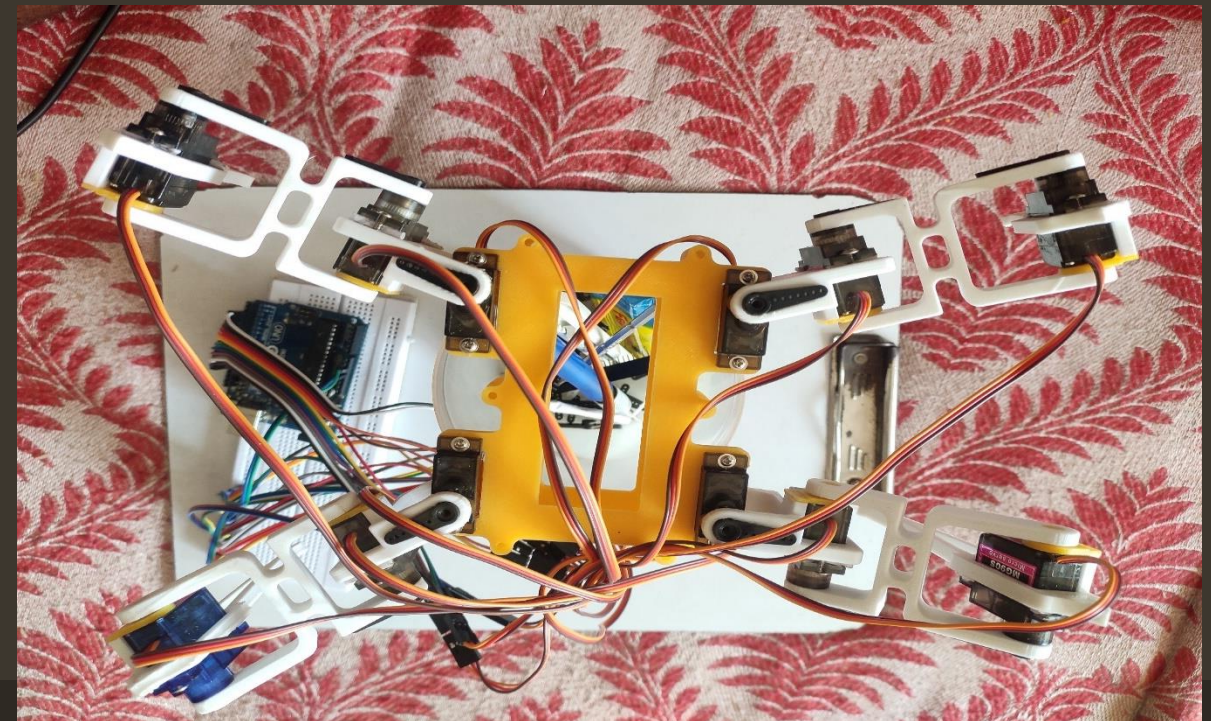
Quadra-Bot using Raspberry Pi

Presented By : Pilli Venkata Praneeth
Roll No : 2019BEC22
Semester : 8th

Submitted to: Dr. Layak Ali
Assistant Professor.

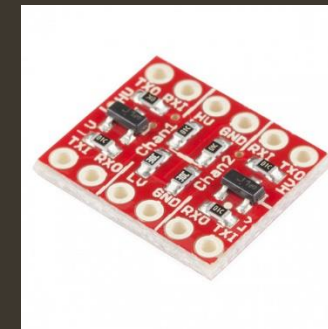
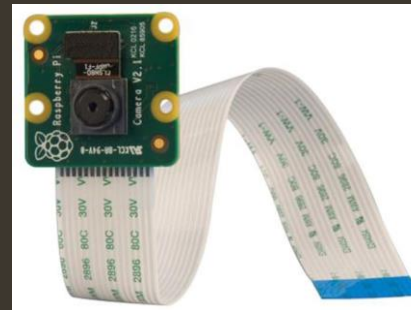
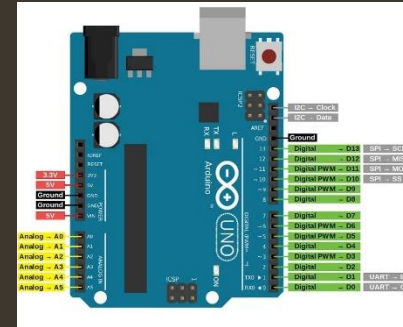
Introduction

- Quadra-Bot is a cutting-edge robot that has been designed to perform various tasks with precision and accuracy. It is equipped with advanced sensors and intelligent algorithms that enable it to navigate through complex environments and carry out its functions efficiently.
- The development of Quadra-Bot has been a significant breakthrough in the field of robotics. Its unique design and capabilities have made it an ideal solution for many industries, including manufacturing, healthcare, and logistics.



Components used :

- Raspberry Pi 3B
- Arduino Uno
- Mg90S servo motors-12
- Pi-Camera module
- logic level shifter
- 3.7v rechargeable li-ion battery
- Buck convertor LM2596



Objective

1. **Mobility:** Four-legged robots are often designed to move over uneven terrain and in environments that are challenging for wheeled or tracked robots.
2. **Stability:** Four-legged robots have a low center of gravity and a wide base, which can provide them with greater stability than other types of robots.
3. **Maneuverability:** Four-legged robots can navigate through narrow spaces and complex environments with greater ease than larger wheeled or tracked robots.
4. **Payload Capacity:** Some four-legged robots are designed to carry payloads, such as sensors or tools, which can be used for a variety of tasks, such as inspection, search and rescue, or transportation.
5. **Versatility:** Four-legged robots can be designed for a variety of applications, including military, industrial, and research, making them a versatile and adaptable technology.

Manufacturing Applications of Quadra-Bot

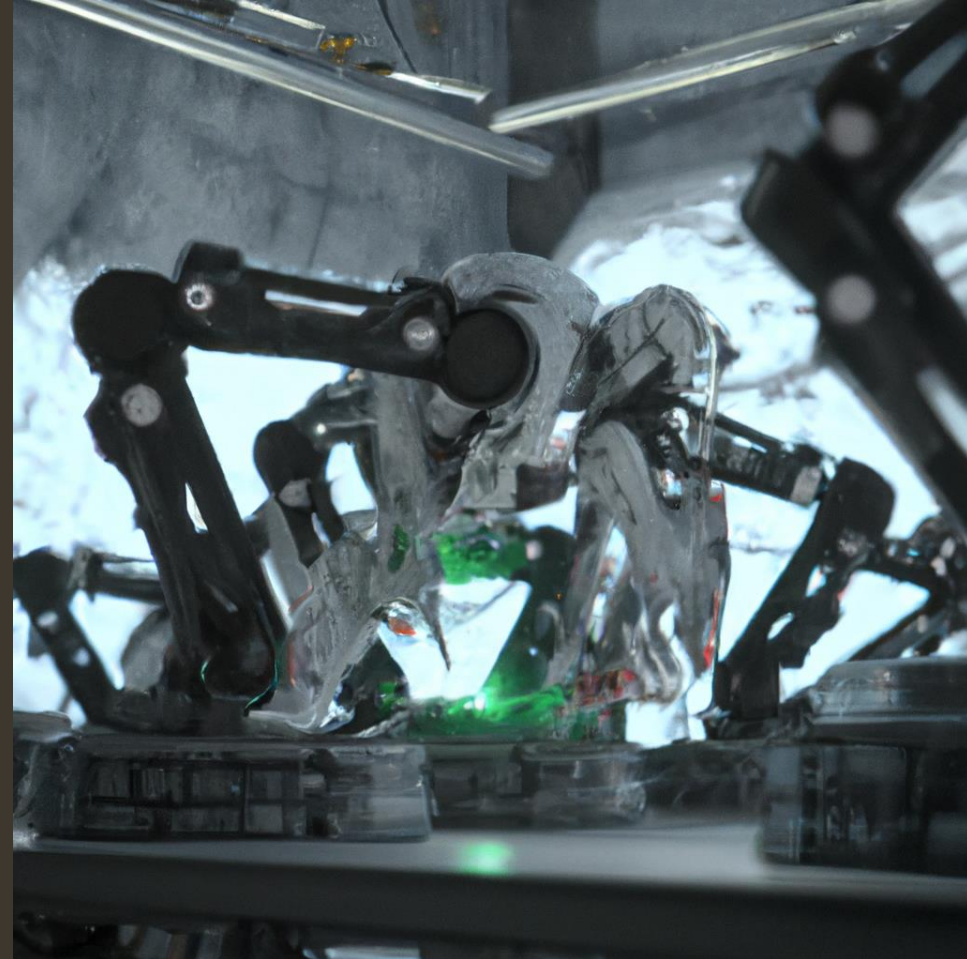
- In the manufacturing industry, Quadra-Bot has proven to be a valuable asset. Its ability to perform multiple tasks simultaneously has increased productivity and reduced production time. It can assemble components, weld, paint, and inspect products with high accuracy and consistency.
- Furthermore, Quadra-Bot's advanced sensors and algorithms allow it to detect defects and errors in the manufacturing process, preventing costly mistakes and ensuring product quality.

Healthcare Applications of Quadra-Bot

- Quadra-Bot has also found applications in the healthcare industry. Its ability to perform delicate procedures with precision and accuracy has made it an ideal tool for surgeries and medical interventions.
- Additionally, Quadra-Bot's advanced sensors and algorithms allow it to monitor patients' vital signs and provide real-time feedback to healthcare professionals, improving patient outcomes and reducing the risk of complications.

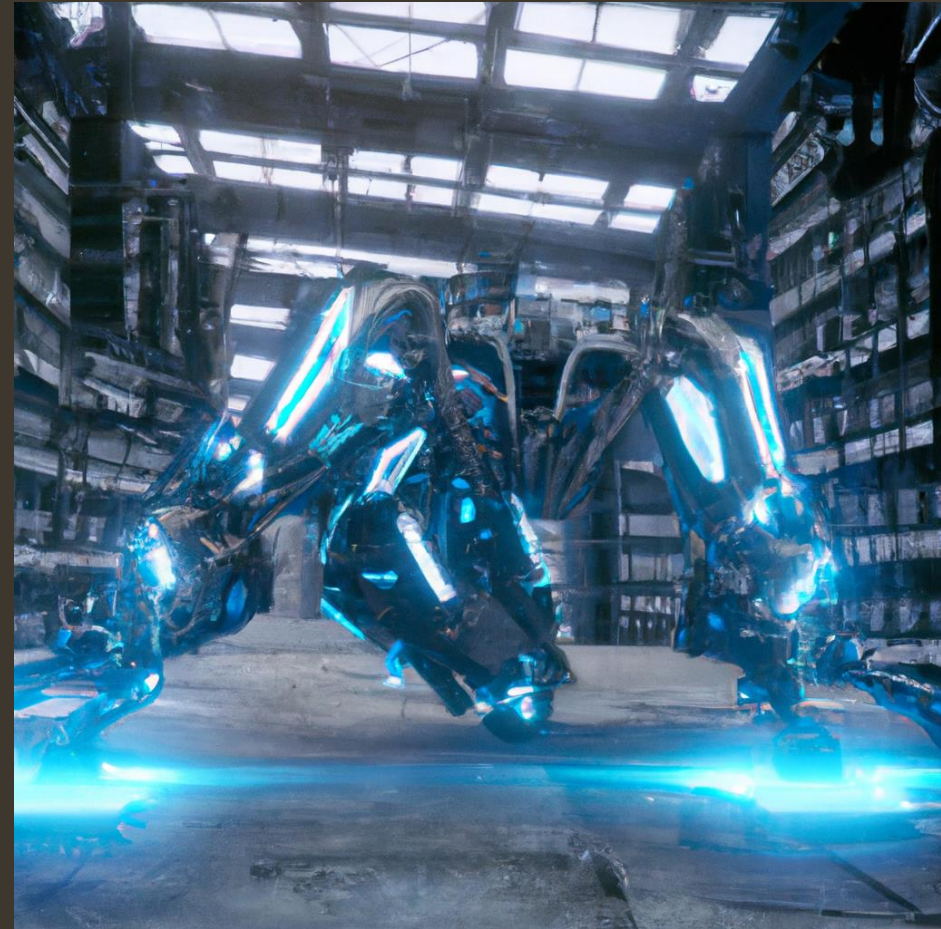
Logistics Applications of Quadra-Bot:

- Quadra-Bot's versatility and precision also make it an ideal solution for logistics and warehouse management. It can efficiently move and stack boxes, pick and pack products, and even load and unload trucks.
- Furthermore, Quadra-Bot's advanced sensors and algorithms allow it to navigate through complex environments and avoid obstacles, reducing the risk of accidents and improving safety in the workplace.

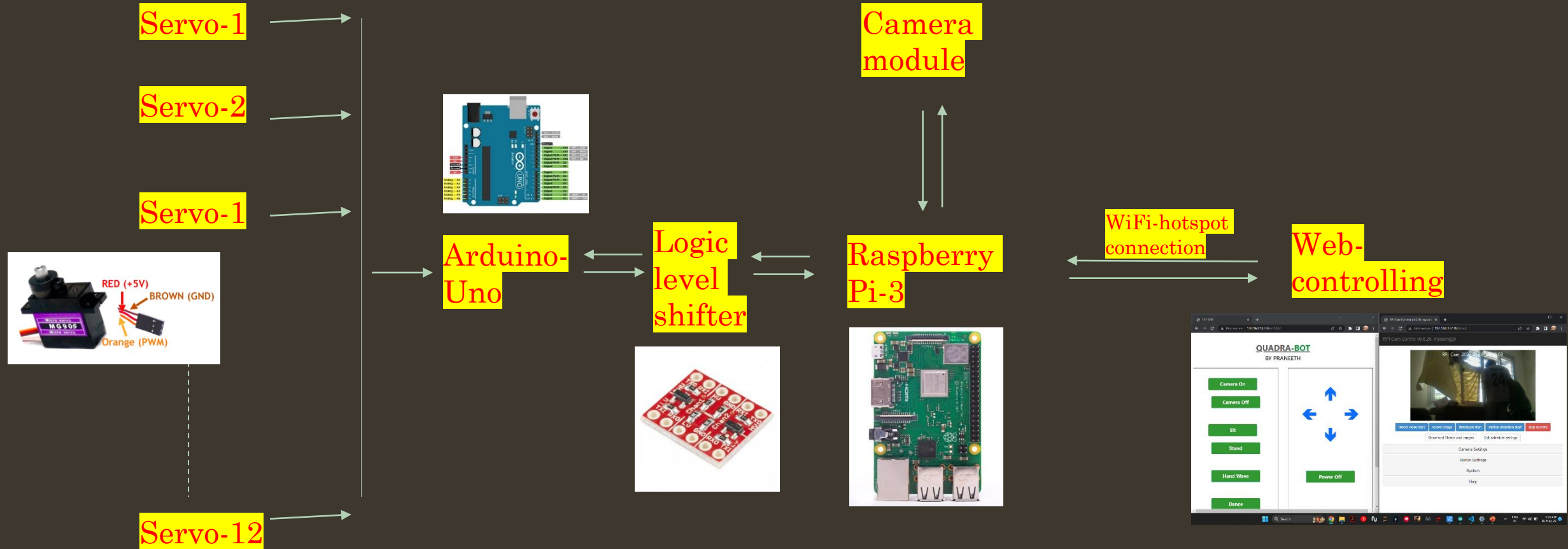


Future Developments of Quadra-Bot

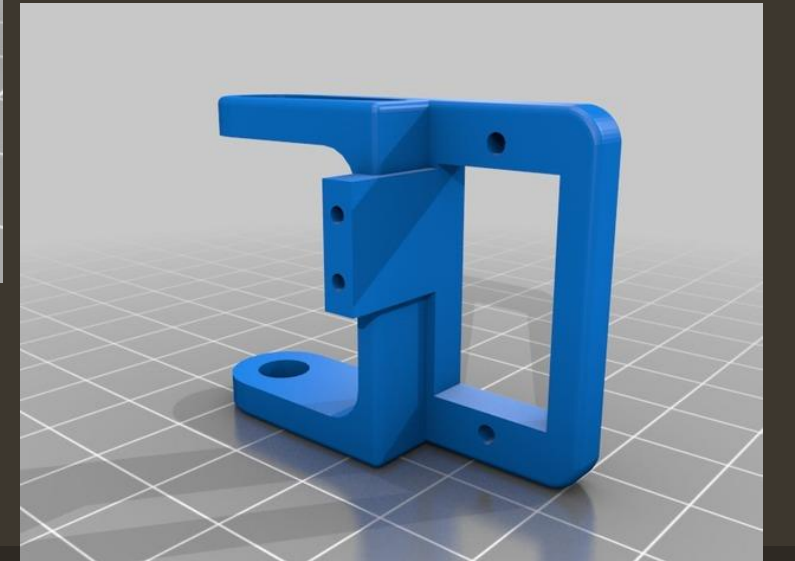
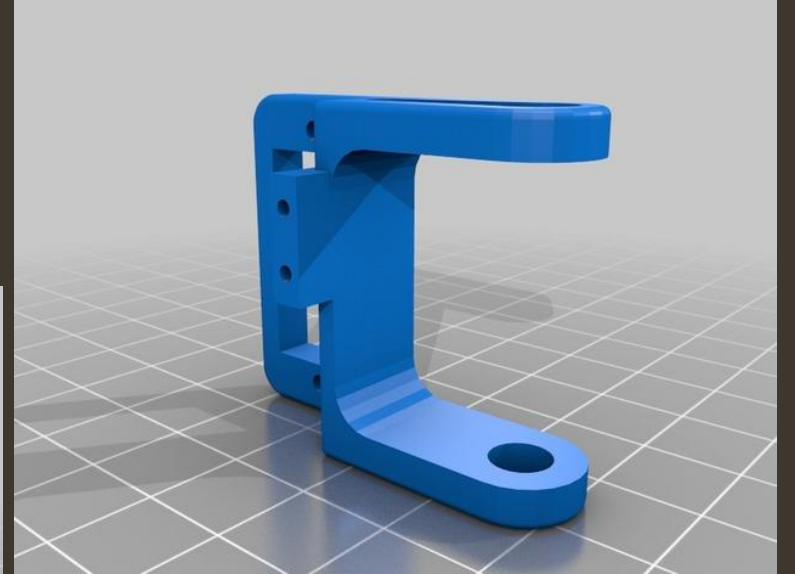
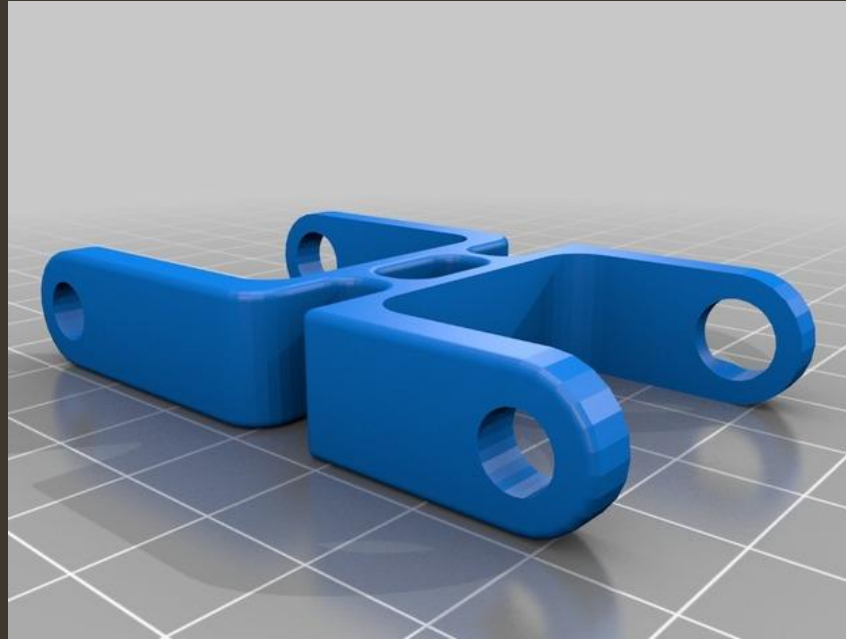
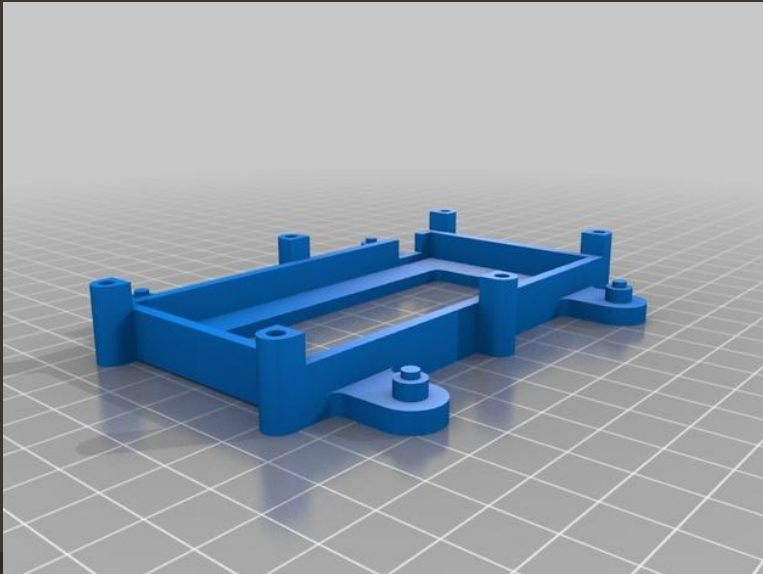
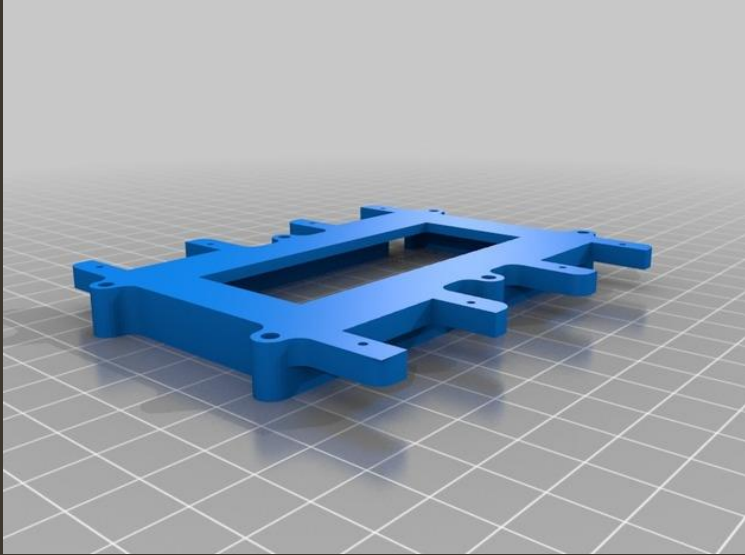
- As technology advances, so will Quadra-Bot. Future developments may include enhanced sensory capabilities, improved algorithms, and increased mobility.
- Additionally, Quadra-Bot may find applications in new industries, such as space exploration and underwater research, where its unique design and capabilities could prove invaluable.



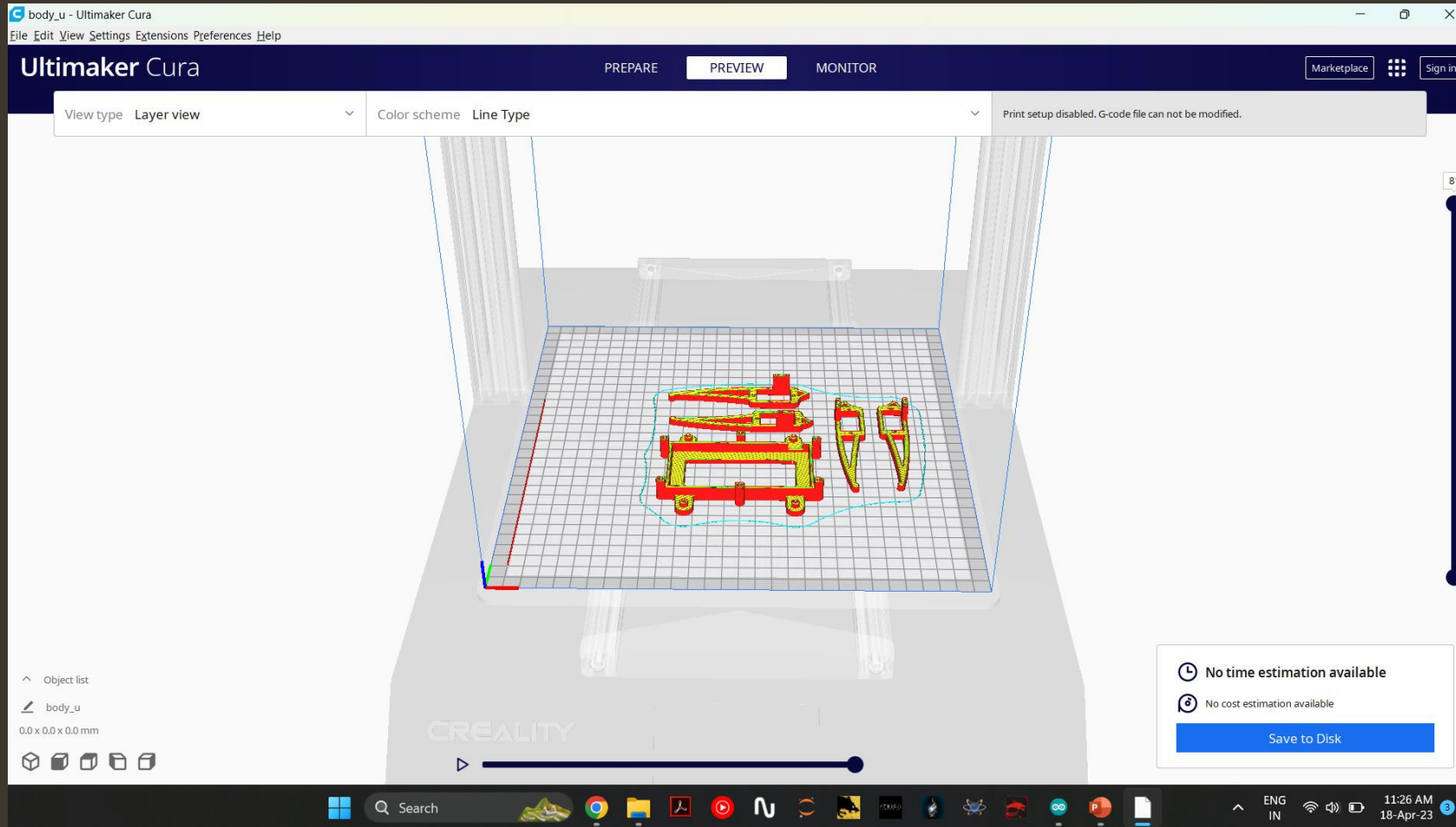
Block Diagram



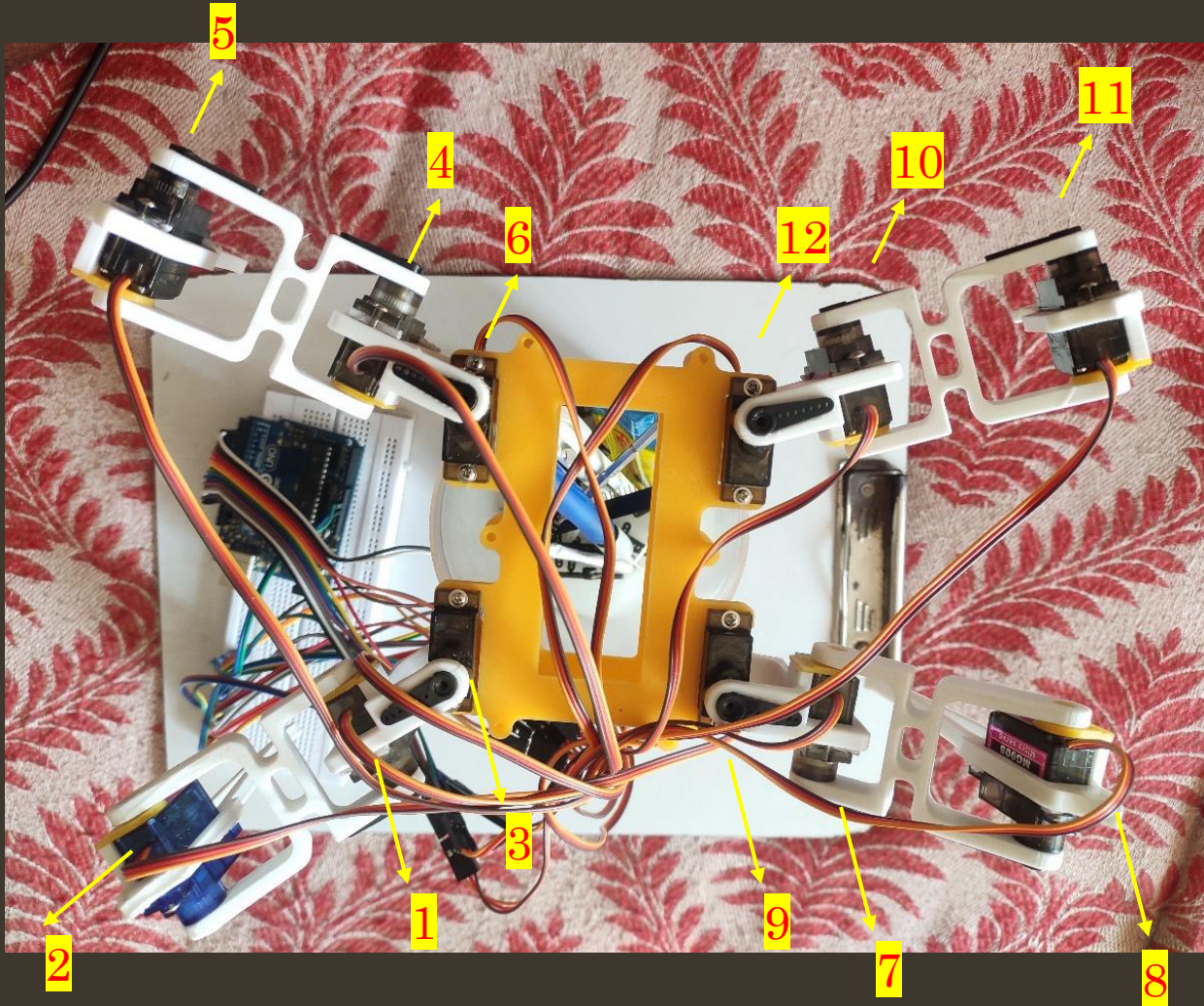
Designing of bot Body



3D printing of designed Body



Implementation



```
Legs | Arduino 1.8.10
File Edit Sketch Tools Help

Legs

#include <Servo.h>

Servo servo[4][3];

//define servos' ports
const int servo_pin[4][3] = { {2, 3, 4}, {5, 6, 7}, {8, 9, 10}, {11, 12, 13} };

void setup()
{
  //initialize all servos
  for (int i = 0; i < 4; i++)
  {
    for (int j = 0; j < 3; j++)
    {
      servo[i][j].attach(servo_pin[i][j]);
      delay(20);
    }
  }
}

void loop(void)
{
  for (int i = 0; i < 4; i++)
  {
    for (int j = 0; j < 3; j++)
    {
      servo[i][j].write(90);
      delay(20);
    }
  }
}
```

Arduino code

```
Legs | Arduino 1.8.10
File Edit Sketch Tools Help

Legs

#include <Servo.h>

Servo servo[4][3];

//define servos' ports
const int servo_pin[4][3] = { {2, 3, 4}, {5, 6, 7}, {8, 9, 10}, {11, 12, 13} };

void setup()
{
    //initialize all servos
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            servo[i][j].attach(servo_pin[i][j]);
            delay(20);
        }
    }
}

void loop(void)
{
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            servo[i][j].write(90);
            delay(20);
        }
    }
}
```

```
SPY-DER_Arduino | Arduino 1.8.10
File Edit Sketch Tools Help

SPY-DER_Arduino FlexiTimer2.cpp FlexiTimer2.h

{
    //start serial for debug
    Serial.begin(9600);
    Serial.println("Robot starts initialization");
    //initialize default parameter
    pinMode(0, OUTPUT);
    set_site(0, x_default - x_offset, y_start + y_step, z_boot);
    set_site(1, x_default - x_offset, y_start + y_step, z_boot);
    set_site(2, x_default + x_offset, y_start, z_boot);
    set_site(3, x_default + x_offset, y_start, z_boot);
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            site_now[i][j] = site_expect[i][j];
        }
    }
    //start servo service
    FlexiTimer2::set(20, servo_service);
    FlexiTimer2::start();
    Serial.println("Servo service started");
    //initialize servos
    servo_attach();
    Serial.println("Servos initialized");
    Serial.println("Robot initialization Complete");
    stand();
}

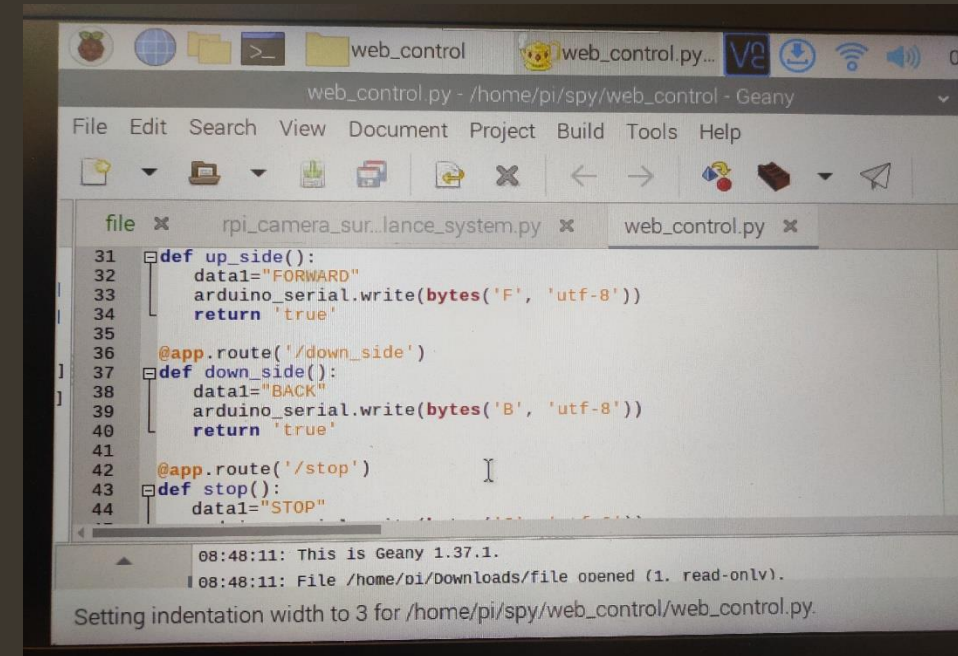
void servo_attach(void)
```

```
SPY-DER_Arduino | Arduino 1.8.10
File Edit Sketch Tools Help

SPY-DER_Arduino FlexiTimer2.cpp FlexiTimer2.h

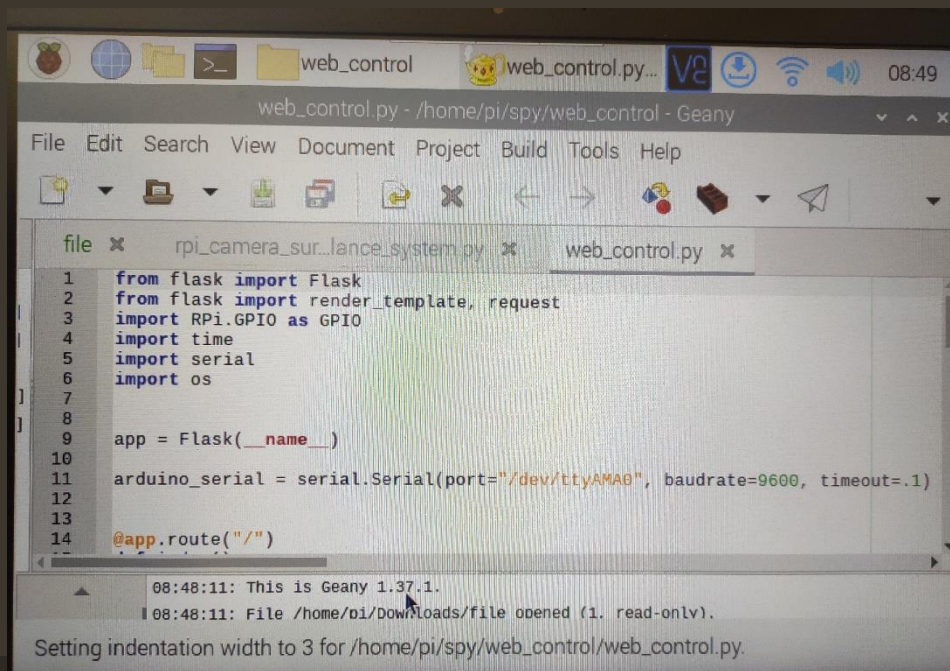
if(Serial.available() > 0)
{
    data = Serial.read();
    Serial.print(data);
    Serial.print("\n");
    if(data == 'f')
    {
        Serial.println("Step forward");
        step_forward(5);
    }
    else if(data == 'b')
    {
        Serial.println("Step back");
        step_back(5);
    }
    else if(data == 'l')
    {
        Serial.println("Turn left");
        turn_left(5);
    }
    else if(data == 'r')
    {
        Serial.println("Turn right");
        turn_right(5);
    }
}
```


Web control using Raspberry- pi 3



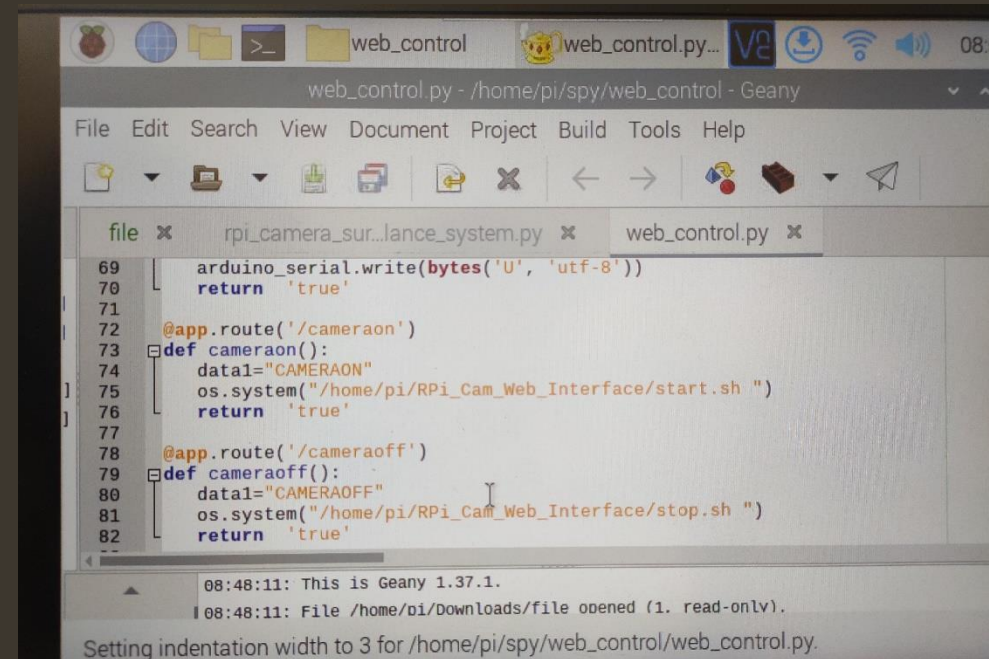
```
31 def up_side():
32     data1="FORWARD"
33     arduino_serial.write(bytes('F', 'utf-8'))
34     return 'true'
35
36 @app.route('/down_side')
37 def down_side():
38     data1="BACK"
39     arduino_serial.write(bytes('B', 'utf-8'))
40     return 'true'
41
42 @app.route('/stop')
43 def stop():
44     data1="STOP"
```

08:48:11: This is Geany 1.37.1.
08:48:11: File /home/pi/Downloads/file opened (1. read-only).
Setting indentation width to 3 for /home/pi/spy/web_control/web_control.py.



```
1 from flask import Flask
2 from flask import render_template, request
3 import RPi.GPIO as GPIO
4 import time
5 import serial
6 import os
7
8
9 app = Flask(__name__)
10
11 arduino_serial = serial.Serial(port="/dev/ttyAMA0", baudrate=9600, timeout=.1)
12
13
14 @app.route("/")
```

08:48:11: This is Geany 1.37.1.
08:48:11: File /home/pi/Downloads/file opened (1. read-only).
Setting indentation width to 3 for /home/pi/spy/web_control/web_control.py.



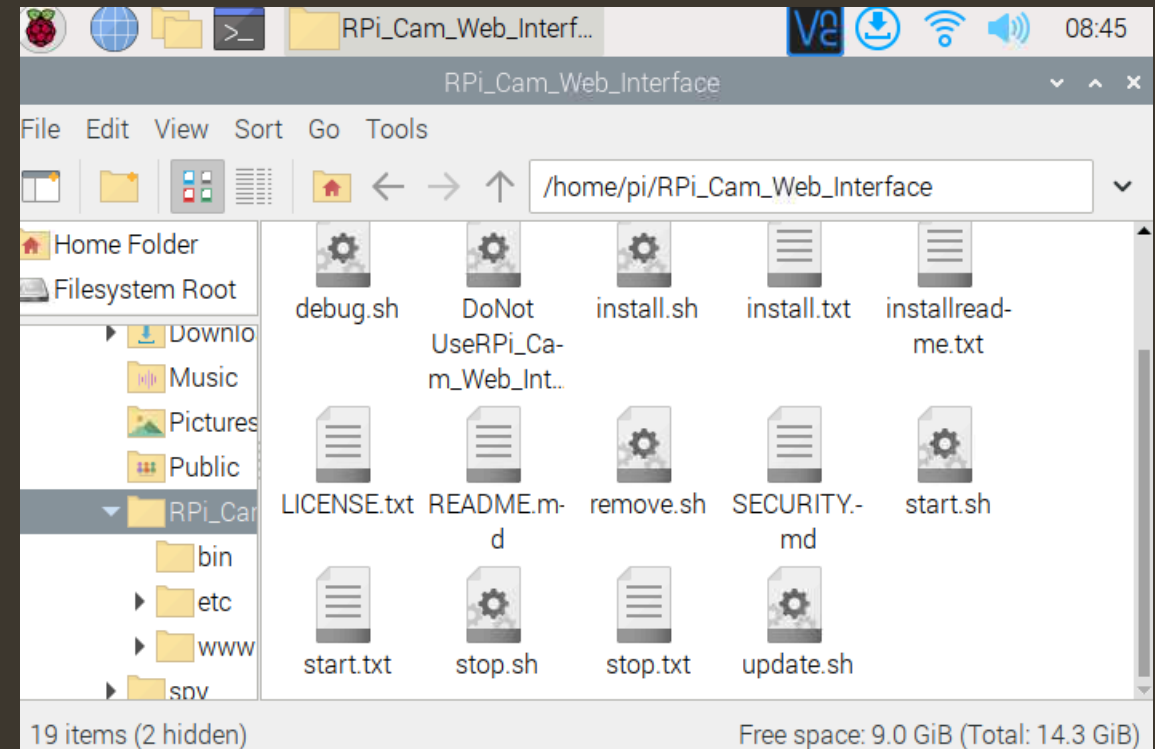
```
69 arduino_serial.write(bytes('U', 'utf-8'))
70 return 'true'
71
72 @app.route('/cameraon')
73 def cameraon():
74     data1="CAMERAON"
75     os.system("/home/pi/RPi_Cam_Web_Interface/start.sh ")
76     return 'true'
77
78 @app.route('/cameraoff')
79 def cameraoff():
80     data1="CAMERAOFF"
81     os.system("/home/pi/RPi_Cam_Web_Interface/stop.sh ")
82     return 'true'
```

08:48:11: This is Geany 1.37.1.
08:48:11: File /home/pi/Downloads/file opened (1. read-only).
Setting indentation width to 3 for /home/pi/spy/web_control/web_control.py.

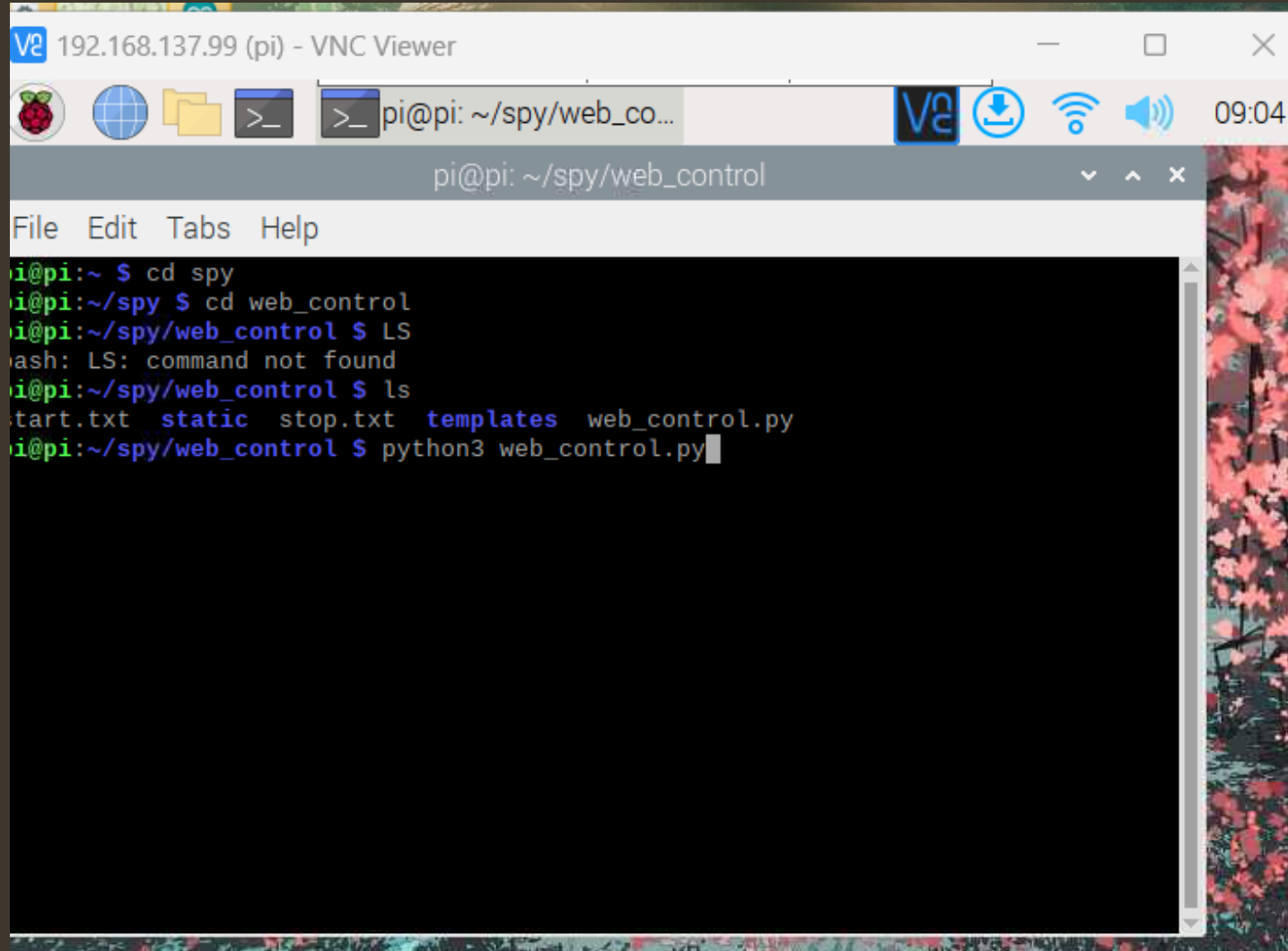
Video surveillance

RPi Cam Web Interface is a web interface for the Raspberry Pi Camera module. It can be used for a wide variety of applications including surveillance, dvr recording and time lapse photography. It is highly configurable and can be extended with the use of macro scripts. It can be opened on any browser (smartphones included) and contains the following features:

- View, stop and restart a live-preview with low latency and high framerate. Full sensor area available.
- Control camera settings like brightness, contrast, ... live
- Record full-hd videos and save them on the sd-card packed into mp4 container while the live-preview continues
- Do timed or continuous video recording with splitting into fixed length segments
- Take single or multiple (timelapse) full-res pictures and save them on the sd-card (live-preview holds on for a short moment)
- Preview, download and delete the saved videos and pictures, zip-download for multiple files



Raspberry Commands



The screenshot shows a VNC Viewer window titled "192.168.137.99 (pi) - VNC Viewer". The window displays a terminal session on a Raspberry Pi. The terminal shows the user navigating to the `~/spy/web_control` directory and running `python3 web_control.py`. The terminal output shows the directory contents: `start.txt static stop.txt templates web_control.py`. The window's taskbar includes icons for the Raspberry Pi, a globe, a folder, a terminal, and a download icon, along with the time 09:04. The desktop background features a floral pattern.

```
pi@pi: ~/spy/web_co...
pi@pi: ~/spy/web_control
File Edit Tabs Help
pi@pi:~ $ cd spy
pi@pi:~/spy $ cd web_control
pi@pi:~/spy/web_control $ LS
bash: LS: command not found
pi@pi:~/spy/web_control $ ls
start.txt static stop.txt templates web_control.py
pi@pi:~/spy/web_control $ python3 web_control.py
```

Interfacing web-control and Pi-camera

The image displays two web browser windows side-by-side, illustrating a web-based control interface for a Raspberry Pi camera system.

Left Window (Control Panel):

- Browser Tab:** SPY-DER
- Address Bar:** Not secure | 192.168.137.99:5010/#
- Page Title:** **QUADRA-BOT** BY PRANEETH
- Control Panel:**
 - Left Column (Buttons):** Camera On, Camera Off, Sit, Stand, Hand Wave, Dance.
 - Right Column:** Four blue directional arrows (Up, Down, Left, Right) and a Power Off button.

Right Window (Live Feed):

- Browser Tab:** RPi Cam Control v6.6.26: mycam@pi
- Address Bar:** Not secure | 192.168.137.99/html/
- Page Title:** RPi Cam Control v6.6.26: mycam@pi
- Live Video Feed:** Shows a person in a white shirt with the number 24 on the back, standing near a window. The timestamp "RPI Cam 2023-05-18 09:05:03" is visible in the top right corner of the feed.
- Control Buttons:** record video start, record image, timelapse start, motion detection start, stop camera.
- Utility Buttons:** Download Videos and Images, Edit schedule settings.
- Settings Menu:** Camera Settings, Motion Settings, System, Help.

The Windows taskbar at the bottom shows the system time as 9:06 AM on 18-May-23, with language set to ENG IN.



Thank
You